UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/875,054	06/07/2001	Roberto R. Chinnici	06502.0311	6050

22852 7590 04/23/2003

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
1300 I STREET, NW
WASHINGTON, DC 20005

EXAMINER
PUNIT, PRAKASH C

ART UNIT	PAPER NUMBER
2175	4

DATE MAILED: 04/23/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/875,054	CHINNICI ET AL.	
	Examiner	Art Unit	
	Prakash C Punit	2175	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☐ Responsive to communication(s) filed on _____ .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-30 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-30 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .
4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: .

DETAILED ACTION

Claim Rejections - 35 USC § 112

1. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

2. Claims 26 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 26 recites the limitation "said general computer language programming call" in line 5. There is insufficient antecedent basis for this limitation in the claim.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ireland et al. (U.S. Patent No. 6,266,666) in view of Jacobs et al. (U.S. Patent No. 6,385,643).

As to claim 1, Ireland et al. teaches a computer implementation method, comprising:

receiving database record information at a client computer system (210) from a database server (230) (see Fig. 2; also see column 2, lines 27-30; also see column 8, lines 52-57);

modifying the database record information at said client computer system (210) using a first computer programming language (see column 6, lines 33-38; also see column 7, lines 48-62; where "first computer language" is read on "Structured Query Language"; the client uses SQL for querying or updating records; and "modifying" is read on "database updates");

transmitting the database record information with modifications (i.e. updates) to an application server (221) (see column 6, line 63 through column 7, line7; the CTS (221) is responsible for handling all transactions that take place between client (210) and the database server (230) including processing);

converting (i.e. from ActiceX requests to TDS protocol) the modifications (i.e. updates), at the application server (221) to (see column 8, line 36-39) and

executing to invoke functions (see column 4, lines 39-44) to cause database record changes at said database server (230) that correspond to the modifications to the database record information (see column 10, lines 26-36; where "modifications" is read on "write data").

Ireland et al. does not expressly disclose the second computer programming language calls of the computer application;

Jacobs et al. discloses the second computer programming language calls of the computer application (see column 2, lines 14-24; where "computer application" is read on "java application");

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include a second computer programming language of a computer application.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because by having a second computer programming language of a computer application, it not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 2, Ireland et al. teaches a method, further comprising the step of determining when a user (210) has completed making changes to said database record information at said client computer system (see column 7, lines 48-55; where "determining when user has completed making changes" is read on "supports full two-phase commit"; also see column 12, line 58 through column 13, line 31; where the CTS determines that the client has completed making changes by looking at the commit statement).

As to claim 3, Ireland et al. teaches a method, wherein said step of transmitting said modifications (i.e. changes) to said application server (221) comprises transmitting to said application server (221) a list of changes made to said database record information (see column 6, lines 29-38; where "one or more packets" is interpreted as several updates to the database).

As to claim 4, Ireland et al. teaches a method, wherein said application server (221) determines the changes (i.e. result set) that have been made to said database record information from said list (see column 8, lines 16-31; where "collection of objects' indicates that several updates are possible and are returned to the client in a result set by the CTS) and converts said changes to functions of said application that cause modifications to a database record of said database server (230) that corresponds to the modifications (i.e. updates) made at said client computer system (see column 10, lines 26-36; where "converts the changes of functions of the application that cause modifications" is read on "CTS converts the native data types to Open server and proceeds to invoke write data").

As to claim 5, Ireland et al. teaches a computer method, wherein said database record information represents at least a subset (i.e. rows) of a table of said database server (230) (see column 8, lines 65-67; also see column 18, lines 45-46) and said step of modifying comprises inserting an element in said subset of said table at said client computer system (see column 13, lines 37-44; also see column 12, lines 59 through column 13, line 16; the client requests modification in the form of an insert statement embedded in the routine).

As to claim 6, Ireland et al. teaches a method, further comprising determining, at said application server (221), subset (i.e. rows) modified at said client computer system (see column 12, line 57 through column 13, line 31; client requests for modification in the form of a routine and sends to CTS); and creating a map to enable inserts into a table of said database server (230) that corresponds to said step of inserting at said client computer system (see column 13, lines 15-

39; the insert command has the information such as insert data, data types, table name and so on for insertion of data).

Ireland et al. does not expressly disclose an EJB object that corresponds to the table.

Jacobs et al. discloses an EJB object that corresponds to the table (see Fig. 5B; where EJB object is shown connected to the database server 509, which implies the EJB objects correspond to the tables stored in the database).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include an EJB object that corresponds to the table.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because the EJB not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 7, Ireland et al. teaches a method, further comprising:

identifying all create methods (see column 9, lines 26-35; also see column 10, lines 57-65; where "objects" is read on "result set");

determining which columns of tables of said database (230) correspond to arguments of identified create methods (see column 10, lines 14-23); and

wherein said step of creating said map comprises mapping said columns to arguments of the create methods that correspond to said columns (see column 9, lines 47-55; also see column

10, lines 25-36; where "mapping of columns" is read on "converting native data type to corresponding ones").

Ireland et al. does not expressly disclose EJB object and sub-objects.

Jacobs et al. discloses EJB object and sub-objects (see Fig. 5B; also see column 2, lines 20-24).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include EJB object and sub-objects.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because having an EJB object and sub-objects not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 8, Ireland et al. teaches a method, further comprising determining, at said application server (221), the location of the insert into said subset (i.e. rows) of said table (see column 17, line 66 through column 18, line 1; where "location of the insert" is read on "populate") and mapping the element inserted (see column 18, lines 1-5; also see column 10, lines 25-32; where CTS converts native data types of the components to open server at the time of populating) into said subset to an argument of the identified create method that is operative to cause said element to be inserted into said table of said database server (see column 9, lines 5-18; where "inserting" is read on "transaction").

As to claim 9, Ireland et al. teaches a method, further comprising executing the identified create method (see column 9, lines 26-33; stub is responsible for executing the methods), at the application server (221), to cause the element to be inserted in said table of said database (see column 12, line 29 through column 13, line 44).

As to claim 10, Ireland et al. teaches a method of interfacing between a client computer system (210) and a database server (230), comprising the following steps:

receiving, at an application server (221), a set of commands from a client computer system to modify a database record of a database server (230) (see column 6, lines 33-38; also see column 7, lines 1-7; where "modify a database record" is read on "client sessions");

at said application server (221), that are operative to insert elements into a database record (see column 7, lines 38-62; where "application instructions" is read on "communication sessions");

to be correlated to certain commands received from said client computer system (210) that indicate the insertion of an element into said database record (see column 8, lines 30-39; also see column 13, lines 36-40);

wherein execution of said selected certain application instructions cause the invocation of a database call (see column 4, lines 39-44) to insert elements into said database record corresponding to said certain commands (see column 10, lines 26-36; where "modifications" is read on "write data").

Ireland et al. does not expressly disclose identifying, enabling and executing selected certain instructions of an application.

Jacobs et al. discloses identifying, enabling and executing selected certain instructions of an application (see column 4, lines 1-13; also see column 8, lines 38-59; it implicitly implies that execution takes place only after the commands have been identified);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include identifying, enabling and executing selected certain instructions of an application.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because by having a execution of application instructions capability, it not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 11, Ireland et al. as modified does not expressly disclose a method, wherein said step of identifying occurs when said application is being configured for execution at said application server.

Jacobs et al. discloses step of identifying occurs when said application is being configured for execution at said application server (see column 4, lines 1-13; also see column 8,

lines 38-59; It is implicitly indicated that first the commands are identified by the server then execution of code takes place);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include identifying, enabling and executing selected certain instructions of an application.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because by having a execution of application instructions capability, it not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 12, Ireland et al. as modified teaches a method, wherein said step of enabling comprises identifying columns of database records that may be operated on by the identified (see column 10, lines 17-20; where "identifying columns" is read on "binding variables to those columns").

Ireland et al. does not expressly disclose enabling certain application instructions.

Jacobs et al. discloses enabling certain application instructions (see column 2, lines 20-25; also see column 4, lines 1-13; where "application instructions' is read on "java objects");

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include enabling certain application instructions.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because by having enabling certain application instructions capability, it not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 13, Ireland et al. as modified teaches a method, further comprising mapping the columns that may be operated on to the identified certain application instructions (see column 18, lines 1-5; also see column 10, lines 25-32; where CTS converts native data types of the components to open server at the time of populating).

As to claim 14, Ireland et al. as modified teaches a method, further comprising updating (i.e. multi-database updates) elements the columns associated with the identified application instructions based on the certain commands (see column 7, lines 53-55).

As to claim 15, Ireland et al. as modified teaches a method, wherein said commands (i.e. calls) are from a different type database access protocol language (i.e. Structured Query Language) than said certain application instructions (i.e. java) (see column 6, lines 33-38).

As to claim 16, Ireland et al. as modified teaches a method, wherein said step of identifying said certain application instructions comprises identifying create methods (see column 9, lines 47-55; where "create methods" is read on "create a new data window").

As to claim 17, Ireland et al. as modified teaches a method, wherein said database record is a table of said database (see column 17, lines 62-65).

As to claim 18, Ireland et al. as modified teaches a method, wherein said create method (see column 9, lines 47-55; where "create methods" is read on "create a new data window") is part that accesses the table.

Ireland et al. does not expressly disclose EJB object.

Jacobs et al. discloses EJB object (see Fig. 5B).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include EJB object.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because having an EJB object not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 19, Ireland et al. as modified teaches a method, wherein identifying said create methods (see column 9, lines 47-55; where "create methods" is read on "create a new data window").

Ireland et al. does not expressly disclose identifying EJB object.

Jacobs et al. discloses identifying EJB object (see Fig. 5B; see column 2, lines 25-33; also see column 10, lines 57-67).

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include identifying EJB object.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because having an EJB object not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 20, Ireland et al. teaches a computer readable medium, operative to serve as a database interface, having instructions which when executed by a computer system perform a method comprising the following steps:

identifying first level software components of an application on an application server (221) (see column 8, line 49 through column 9, line 4; also see column 9, lines 18-25);

exposing the first level software components in association with operations (see column 10, lines 13-31; where "exposing" is read on "invoke write data");

mapping modification commands received (see column 10, lines 13-23; also see column 18, lines 1-9; where "modifications" is read on "component's language"), at the application server (221), from a client computer system (210) to the identified first level components that correspond to the modification commands (see column 9, lines 26-35); and

to update said database in accordance with the modifications received from the client computer system (see column 7, lines 52- 62).

Ireland et al. does not expressly disclose sub-level software components for accessing data input fields of a database.

Jacobs et al. discloses sub-level software components for accessing data input fields of a database (see column 8, line 66 through column 9, line 8; also see Fig. 5b; where "sub-level software component" is read on "EJB");

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include sub-level software components for accessing data input fields of a database.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because by having sub-level software components for accessing data input fields of a database, it not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

Ireland et al. does not expressly disclose executing the identified sub-level software components.

Jacobs et al. discloses executing the identified sub-level software components (see column 8, lines 55-64; it implicitly implies that execution takes place only after the commands have been identified);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include executing the identified sub-level software components.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because having sub-level software executing capability, it not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 21, Ireland et al. as modified does not expressly teach a method, wherein said first level software components is an EJB object and said sub-level software components are sub-objects of said EJB object.

Jacobs et al. discloses first level software components is an EJB object and said sub-level software components are sub-objects of said EJB object (see Fig. 3C and Fig. 5B; also see column 2, lines 20-24);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include EJB object and sub-objects.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because having an EJB object and sub-objects not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 22, Ireland et al. as modified teaches a method, wherein said step of executing comprises executing software components that produce SQL calls to said database to modify said database (see column 10, lines 13-35; where "SQL calls" is read on "Interface calls").

As to claim 23, Ireland et al. as modified teaches a method, wherein said modification commands are from an application (i.e. Powerbuilder) designed for direct access to a relational database (230) (see column 10, lines 57-65).

As to claim 24, Ireland et al. as modified does not expressly teach a method, wherein said identified first level and sub-level software components are EJB components.

Jacobs et al. discloses identified first level and sub-level software components are EJB components (see Fig. 3C and Fig. 5B; also see column 2, lines 20-24);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include first level and sub-level EJB components.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because having first level and sub-level EJB components not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 25, Ireland et al. teaches a computer implemented method for interfacing between a client computer system and a database server, comprising the following steps:

receiving, at the application server (221), the result of a query request as modified by a client computer system (210) (see column 6, lines 33-38; also see column 7, lines 1-7; where "modified" is read on "client sessions");

determining, at the application server (221), the modifications made to the result of the query request (see column 7, lines 47-55; where "modifications" is read on "transactions") ;

converting (i.e. from ActiceX requests to TDS protocol), at said application server (221), the modifications from a first programming language (see column 8, line 32-39) into; and

executing command to produce a database protocol (i.e. InterORB Protocol) command to modify (i.e. write) a database record to correspond to the query request as modified by the client computer system (210) (see column 10, lines 24-43; also see column 12, line 56 through column 13, line 44; where runCommands() determines what type of command the client has sent).

Ireland et al. does not expressly disclose a general computer programming language command for accessing a database;

Jacobs et al. discloses a general computer programming language command for accessing a database (see column 2, lines 14-24; where "general computer programming language" is read on "java");

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include a general computer programming language command for accessing a database.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because by inclusion of a general computer programming language command for accessing a database, it not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 26, Ireland et al. as modified does not expressly disclose said general computer language programming call is an Enterprise Java Bean (EJB) call.

Jacobs et al. discloses said general computer language programming call is an Enterprise Java Bean (EJB) call (see Fig. 5B, element 552);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include general computer language programming call is an Enterprise Java Bean (EJB) call.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because an

Enterprise Java Bean (EJB) call not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 27, Ireland et al. as modified teaches a method, wherein the database protocol command is a Structured Query Logic (SQL) call (see column 6, lines 33-38; where "SQL call" is read on "client calls in SQL").

As to claim 28, Ireland et al. as modified teaches a method, wherein the application server (221) receives the database call (i.e. requests for data) from a client computer system (210) (see column 8, lines 49-67).

As to claim 29, Ireland et al. as modified teaches a method, further comprising generating a database call (i.e. interface calls) to a database (see column 10, lines 17-20; also see column 10, lines 58-65).

Ireland et al. as modified does not expressly disclose, executing the general computer language programming call.

Jacobs et al. discloses executing the general computer language programming call (i.e. EJB call) (see Fig. 5B, element 552; also see column 8, lines 55-59; where thread manager software component 364 is responsible for executing the programming calls);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include executing the general computer language programming call.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because a general computer language programming call not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

As to claim 30, Ireland et al. as modified teaches a method, further comprising:

generating database calls to a database (see column 10, lines 17-20; also see column 10, lines 58-65);

analyzing the components (i.e. component models) to determine the correspondence between database elements (i.e. columns) and the elements of the components that access the database elements (see column 8, lines 32-47; also see column 10, lines 13-23); and

creating a map that identifies the correspondence (see column 18, lines 1-5; where the APIs are mapped to open interfaces implies correspondences are identified).

Ireland et al. as modified does not expressly disclose, executing the general computer language programming calls.

Jacobs et al. discloses executing the general computer language programming calls (i.e. EJB call) (see Fig. 5B, element 552; also see column 8, lines 55-59; where thread manager software component 364 is responsible for executing the programming calls);

Therefore, it would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. to include executing the general computer language programming calls.

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Ireland et al. by the teachings of Jacobs et al., because a general computer language programming call not only supports a variety of computer programming models, also enables (1) enhanced fault tolerance, (2) efficient scalability (3) effective load balancing and (4) session concentration control (see Jacobs et al., column 3, lines 51-60).

Conclusion

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

The following patents are cited to further show the state of art with respect to database access systems in general:

U.S. Patent No. 6,266,666 to Ireland et al. - teaches business applications

U.S. Patent No. 6,385,643 to Jacobs et al. - teaches Enterprise Java Beans (EJB)

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Prakash Punit whose telephone number is (703) 305-5914. The examiner can normally be reached on Mondays – Fridays from 9:45 am to 6:15 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Dov Popovici, can be reached on (703) 305-3830. The fax numbers of the group is (703) 746-7239.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

Prakash Punit
Patent Examiner
Au 2175

April 18, 2003

SAFET METJAHIC
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100